REMARKS/ARGUMENTS

The present Amendment is in response to the Final Office Action having a mailing date of April 17, 2006. Claims 1-26 are pending in the present Application. Applicant has amended claims 1-3, 6, 8-10, 13, 15, 16, and 25. Consequently, claims 1-25 remain pending in the present Application.

Applicant has amended claims 1-3, 6, 8-10, and 13 to remove alphanumeric designations for steps and instructions. This amendment is seen by Applicant as broadening or cosmetic, and as such, is not subject to the prosecution history estoppel imposed by Festo. For the record, Applicant points out that the Supreme Court in Festo noted that a cosmetic amendment would not narrow the patent's scope and thus would not raise the estoppel bar.

Applicant has also amended claims 1, 8, and 25 to recite that the output(s) of the column function are stored in the relational database. Support for the amendment can be found in the Figure 3, step 106 and the accompanying discussion in the specification, for example page 8, lines 1-7. Applicant has also amended claims 1, 8, 15, and 25 to recite that the row(s) are received as an argument for a generalized scalar function. Support for the amendment can be found in Figure 3, step 102 and the accompanying discussion in the specification, page 7, lines 3-20. Furthermore, Applicant respectfully submits that the scope of claims 1, 8, 15, and 25 is not narrowed and that no new matter is added. Applicant has also amended claims 2, 9, and 16 to recite that the row is provided entry by entry to the column function. Support for the amendment can be found in the specification, page 9, line 14-pate 10, line 8. Accordingly, Applicant respectfully submits that no new matter is added.

In the above-identified Office Action, the Examiner objected to the specification under 37 C.F.R. 1.71 as failing to provide an adequate description of the invention. In particular, the

Examiner stated that the specification fails to disclose "the actual, practical steps for forming the claimed abstract generalized scalar function, activating the claimed abstract generalized scalar function[,] initializing the first entry, evaluating each entry and finalizing the lat entry of the at least one raw [sic], such that the simulating of a column environment will produce a useful, concrete, and tangible result."

Applicant respectfully disagrees with the Examiner's objection to the specification. Applicant respectfully submits that the recited generalized scalar function, the column function that initializes, evaluates, and finalizes the resultant, and the cooperation between the generalized scalar function and the column function are sufficiently disclosed in the specification. As described in the specification, the generalized scalar function provides row data to the column function so that the column function can utilize the data (i.e. simulates a column environment), while the column function operates in a conventional manner.

The generalized scalar function is adequately described in the specification and depicted in the drawings. Moreover, the operation of the generalized scalar function in conjunction with the column function is adequately described. As described in the specification, and cited in prior responses, one embodiment of the generalized scalar function obtains the data in a row and provides the data, entry by entry, to the column function. Specification, page 9, line 14-page 10, line 8. As expressly stated in the specification, the generalized scalar function fetches a row that has been specified as an argument for the generalized scalar function. Specification, page 9, lines 12-15. The generalized scalar function provides a single entry from the row that has been fetched to a column function, which operates on the entry. Specification, page 9, lines 15-22. See also Specification, page 8, line 10-page 9, line 2. It is determined whether there are additional entries in the row and, if so, these are also provided entry-by-entry to the column function. Specification,

page 10, lines 1-8. Moreover, the specification describes entries as corresponding to intersections of rows and columns (e.g. entry 11 in Figure 1), rather than entire rows or other entities. Specification, page 1, lines 7-14. Because the column function receives the data entry by entry, the column function can operate on the data from the row. Specification, page 9, lines 1-4. Thus, the operation of a generalized scalar function (fetch rows, provide entries in the row to the pre-existing column function) and the cooperation of the generalized scalar function with a conventional column function (column function operates on data provided by the generalized scalar function) are adequately described. Moreover, these operations performed by the generalized scalar function are *depicted* in Figure 4, items 152, 154, 156, 162, 164, and 170. The column function carries out its operations in a conventional manner by performing the appropriate ones of the initialization, evaluation, and finalization phases. Specification, page 9, line 18-page 10, line 6 and Figure 4, items 158, 160, and 166. More specifically, the specification states:

> Referring back to Figures 1 and 4, a row 6, 7 or 8 of the table 1 is fetched, via step 154. **A first entry of the row 6, 7 or 8 is provided to the conventional column function, via step 156.** Thus, steps 154, 156 and 164 (discussed below) are used to simulate the column environment for the rows 6, 7 or 8 that is input as an argument to the generalized scalar function. An initialization phase for the conventional column function is carried out, via step 158. Once the initialization phase is performed or if it is determined that the entry provided is not the first entry, then an evaluation phase is performed, via step 160. Thus, the operations necessary for the conventional column function to provide an output are performed in step 160. Step 160 might include adding the data in the entry to a running sum or determining whether the data in the entry is the minimum or maximum encountered. It is determined whether the entry is the last entry in the row 6, 7 or 8, via step 162. **If not, the next entry in the row is provided to the conventional column function, via step 164.** The method 150 then returns to step 160 to the evaluation phase for subsequent entries. If the entry is the last in the row 6, 7 or 8, then the conventional column function enters it finalization phase, via step 166 and returns an output, via step 168. It is then determined whether there are any additional arguments in the generalized scalar function, via step 170. If so, step 152 is returned to so that the next row can be fetched. Otherwise, the method 150 terminates.

Specification, page 9, line 14-page 10, line 8 (emphasis added). Thus, providing individual entries to the column function (steps 156 and 164) and using the column function (steps 158, 160, 166, and 168) to perform functions on the entries are described. Consequently, Applicant respectfully submits that the discussion previously added regarding providing the row entry-by-entry to the column function did have support in the specification. Consequently, the specification not only describes with specificity an embodiment of the generalized scalar function, but also describes and depicts the operation of this embodiment of the generalized scalar function in conjunction with the conventional column function.

Moreover, an enabling description of the *preexisting* column function is also provided in the specification. As discussed in previous Amendments, the column function is a conventional, preexisting column function. Specification, page 2, line 14-page 3, 12 and page 10, lines 9-13. See also, Paragraph 4, Declaration by Jason Cu (submitted with Amendment filed September 30, 2003). Moreover, as described in the specification, the initialize, evaluate, and finalize are phases used by conventional, preexisting column functions. Specification, page 3, lines 13-19. Consequently, the described column function is a conventional function operating in a conventional manner employing conventional phases. Applicant respectfully submits that one of ordinary skill in the art would readily understand the use of such a *conventional* function and that the phases of the *conventional* column function are adequately described in the specification. Accordingly, Applicant respectfully submits that the specification complies with 37 C.F.R. 1.71, providing an adequate description of the invention.

The Examiner also rejected claims 1-26 under 35 U.S.C. § 101 because the claimed invention is directed to non-statutory subject matter. In so doing, Examiner indicated that the "'allowing' phrase repeatedly recited in independent claims 1, 8, and 15 does not cause any

functionality to occur in the computer system. This is demonstrated by the absent recitation of any code or steps for causing a computer to do anything." The Examiner indicated that in an interview, the "application attorney. . . on record indicated that the instant invention deals with reversing a matrix (or a table)."

Applicant respectfully but vehemently disagrees with the Examiner's rejection. First, Applicant strongly disagrees that the "application attorney" indicated that the instant invention deals with reversing a matrix or table. In the interview, it became apparent that the Examiner was having difficulty understanding functioning of the present invention. Thus, in order to facilitate the Examiner's understanding of the present invention, Applicant's attorney stated that the present invention could be thought of as transposing the matrix in that rows could be operated on by a function that normally operates on columns. In other words, the present invention allows rows to be used (by a column function) as though the rows were columns. Application attorney further stated during the interview and in remarks subsequently filed that this analogy was made in order to aid the Examiner in understanding the present invention, not to characterize the operation of the present invention. As the application attorney has repeatedly stated, the present invention recited in varying scope in claims 1-26 did not function to transpose a matrix. Moreover, Applicant respectfully submits that even if the present invention did transpose a matrix or table in a database system, the present invention would provide a function in a computer system. This function would be to operate to change form or location of stored data.

Applicant also disagrees with the Examiner's indication that the "allowing" phrases do not cause any functionality in a computer system. However, to facilitate prosecution, Applicant has amended claims 1, 8, 15, and 25 to recite "receiving" the at least one row as an argument for a generalized scalar function. Moreover, the independent claims positively recite the "simulating",

"performing", and "storing" functions. For example, in claim 1, the "simulating" step simulates a column environment. As discussed in the specification, the simulating might include providing the row data entry-by-entry to the column function. Furthermore, claim 1 recites the step of "performing the column function on the at least one row to provide at least one output." As a result, the computer system would implement the column function, obtain an output, and *store the output*. Consequently, the method recited in claim 1, as well as the analogous computer-readable media and system recited in claims 8, 25 and 15, do provide functionality in a computer system. As such, the claimed invention is directed to statutory subject matter. Consequently, Applicant respectfully submits that the Examiner's rejection under 35 U.S.C. § 101 is incorrect.

The Examiner also rejected claims 1-26 under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to point out and distinctly claim the subject matter that the Applicant regards as the invention.

Applicant respectfully disagrees with the Examiner's rejection. As recited in independent claims 1, 8, and 15, the generalized scalar function is used to simulate the column environment for the row. The generalized scalar function is described in the specification and depicted in the figures. See, for example, Specification, page 8, line 10-page 9, line 2. See also, page 9, line 12-page 10, line 8. In particular, one embodiment of the generalized scalar function is described as fetching a row, providing the first entry in the row to the column function, and providing subsequent entries in the row to the column function entry by entry. Specification, page 9, lines 12-18 and Figure 4, items 154, 156, 162, and 164. Thus, the form of one embodiment of the generalized scalar function is essentially depicted at a flow chart level in a portion of Figure 4 and described in the accompanying discussion. In addition, the cooperation of this embodiment of the generalized scalar function with the column function is described and depicted. Specification, page

9, lines 12-18; page 10, lines 1-8 and Figure 4. The specification even states that in one embodiment: "the steps 154 [fetch row], 156 [provide first entry in row to column function], and 164 [provide next entry in row to column function] . . . **are used to simulate the column environment** for the rows 6, 7, or 8 that is input as an argument to the generalized scalar function." Specification, page 9, lines 16-18. Consequently, when read in light of the specification, Applicant respectfully submits that the recited generalized scalar function is clear and definite.

With respect to claims 6, 13, and 20, as previously discussed, the column function carries out its operations in a conventional manner by performing the appropriate ones of the initialization, evaluation, and finalization phases. Specification, page 9, line 18-page 10, line 6 and Figure 4, items 158, 160, and 166. Thus, Applicant respectfully submits that these phases are well understood by those of ordinary skill in the art. See also, Declaration by Jason Cu (submitted with Amendment filed September 30, 2003). Consequently, the recited column function is also clear and definite. Accordingly, Applicant respectfully submits that independent claims 1, 8, and 15 are clear and definite.

The Examiner also rejected claims 1-21 under 35 U.S.C. § 102 as being unpatentable over U.S. Patent No. 6,289,336 (Melton).

Applicant respectfully traverses the Examiner's rejection. As a preliminary manner, Applicant notes that in an Office Action dated March 30, 2005, the Examiner expressly stated that Melton "fails to disclose a specific instance of a generalized scalar function linked to a column function (or running and moving sequence function) as specified by a user." Consequently, the Examiner had previously cited another reference in order to remedy the defects of Melton. Thus, despite the Examiner's review of the record, Applicant respectfully submits that as the Examiner

has previously recognized, Melton fails to teach or suggest the method, system, and computer-readable medium recited in claims1-26.

Applicant also respectfully disagrees with the Examiner's rejection for at least the following reasons. Claim 1 recites a method for utilizing a column function for a relational database in a structure query language (SQL) environment. The method recited in claim 1 includes receiving the at least one row as an argument for a generalized scalar function and simulating a column environment for the at least one row using the generalized scalar function to allow the at least one row to be provided to the column function as though the at least one row was a column. The method recited in claim 1 further recites performing the column function on the at least one row to provide at least one output and storing the output(s). Claims 8 and 25 and claim 15 recite analogous computer-readable media and system.

Thus, using the method, computer-readable medium and system recited in claims 1, 8, 15, and 25, the pre-existing column function can be reused to work on row data. As a result, the resources that would otherwise be used in rewriting, testing, and debugging a row function that performs the operations of the column function are saved. Specification, page 9, lines 1-6; page 10, lines 9-13. Stated differently, the method, computer-readable medium, and system recited in claims 1, 9, and 15, respectively, can be used to extend the ability of the pre-existing column function to operate on row data without expending significant additional resources.

In contrast, the cited portions Melton fails to teach of suggest the use of the recited generalized scalar function in conjunction with a (pre-existing, conventional) column function. Instead, the cited portions of Melton describe a specific set of row functions that are written. Melton expressly states that the sequence functions described in Melton "are functions that operate on ordered sets of rows and that require knowledge of or access to past values." Melton,

col. 1, lines 13-16. Thus, Melton describes row functions that are written to search previously accessed rows or offsets. Melton, col. 2, lines 10-45. Stated differently, the functions described in the cited portion of Melton are for rows, not columns, and to perform specific operations on these rows. Applicant can find no indication in the cited portions of Melton that the functions of Melton are used in conjunction with pre-existing column functions. The cited portions of Melton are also devoid of mention of utilizing a generalized scalar function to simulate a column environment so that the row data appears to the column function as a column. For example, the cited portion of Melton also does not describe fetching a row and providing the row data to the column function as if the row was a client. Thus, the cited portions of Melton fails to teach or suggest using a generalized scalar function to allow the row(s) to be provided to the column function as though the at least one row was a column in conjunction with a column function that performs its operation in a conventional manner.

Moreover, Applicant respectfully disagrees that an SQL compiler, such as described in Melton can teach or suggest the recited generalized scalar function. This SQL compiler of Melton is merely for "compiling Offset sequence functions." Melton, col. 4, lines 1-2. Thus, the SQL compiler merely a set of blocks by which the SQL executor can execute the previously discussed sequence functions. Melton, col. 4, lines 12-14. Thus, as previously argued, Melton merely describes a system in which sequence functions are specifically written to perform operations on rows, the sequence functions are converted into executable instructions during compilation, then executed. Melton, col. 2, lines 4-45. Consequently, the cited portions of Melton fail to teach or suggest the method, computer-readable medium and system recited in claims 1, 8, 15, and 25. Accordingly, Applicant respectfully submits that claims 1, 8, 15, and 25 are allowable over the cited references.

Claims 2-7 and 22, claims 9-14 and 23, claims 16-21 and 24 and claim 26 depend upon independent claims 1, 8, 15, and 25, respectively. Consequently, the arguments herein apply with full force to claims 2-7, 9-14, 16-24, and 26. Accordingly, Applicant respectfully submits that claims 2-7, 9-14, 16-24, and 26 are allowable over the cited references.

Applicant's attorney believes that this application is in condition for allowance. Should any unresolved issues remain, Examiner is invited to call Applicant's attorney at the telephone number indicated below.

Respectfully submitted,
SAWYER LAW GROUP LLP

June 19, 2006
Date

/Janyce R. Mitchell/ Reg. No. 40,095
Janyce R. Mitchell
Attorney for Applicant(s)
(650) 493-4540